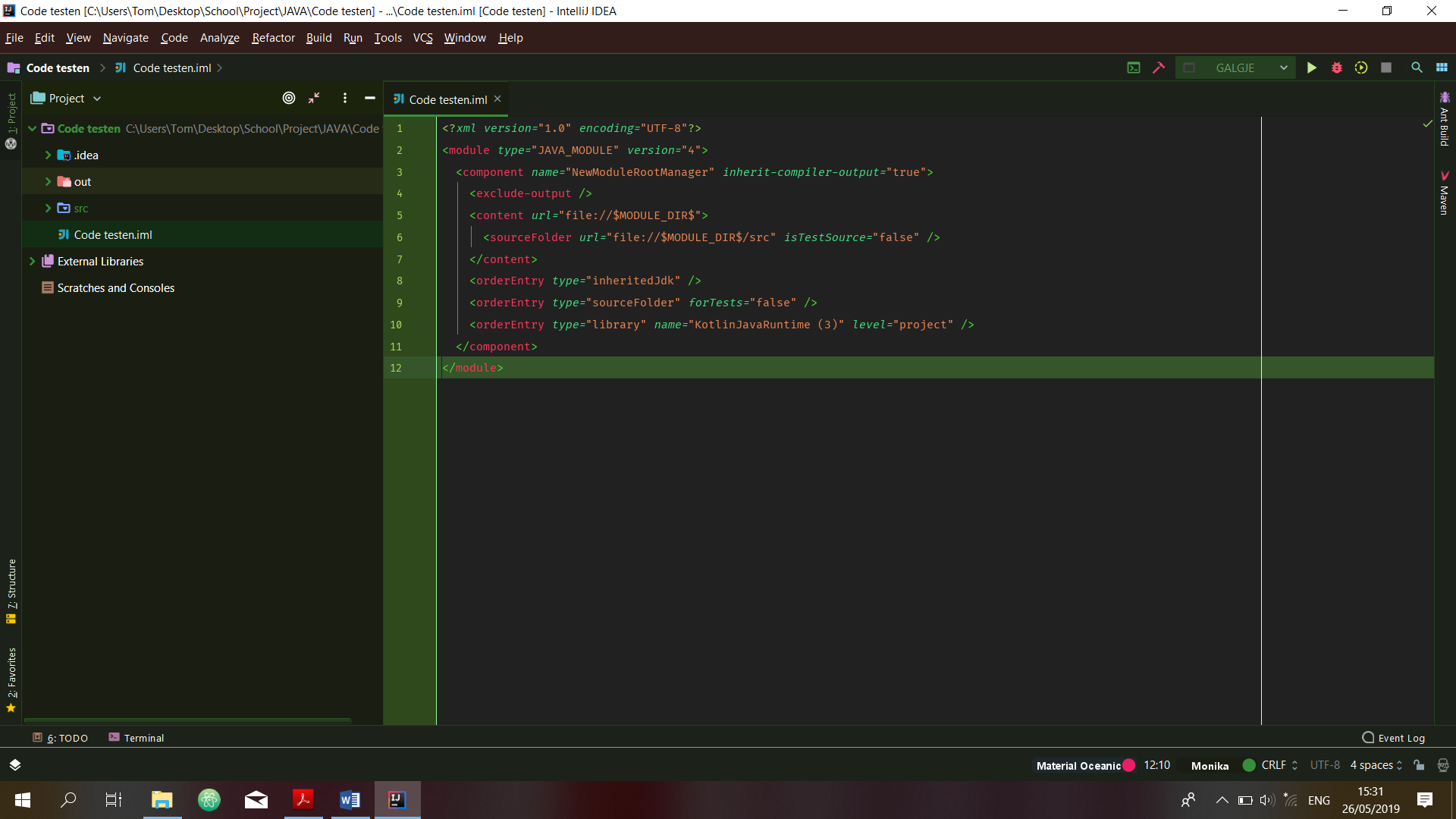
FO, TO en US

DG2 eidopdracht



Inhoud

[Functioneel ontwerp 3](#_Toc9779469)

[Opdrachtgever 4](#_Toc9779470)

[Algemene eisen 5](#_Toc9779471)

[Eindgebruiker(s) 6](#_Toc9779472)

[Wat zijn de verwachtingen 7](#_Toc9779473)

[Technisch Ontwerp 8](#_Toc9779474)

[Functionele eisen 9](#_Toc9779475)

[Toelichten 10](#_Toc9779476)

[Eindproduct 11](#_Toc9779477)

[Systeemonderdelen 12](#_Toc9779478)

[Kwaliteitseisen 13](#_Toc9779479)

[Afhankelijkheden 14](#_Toc9779480)

[Structuren 15](#_Toc9779481)

[User-stories 16](#_Toc9779482)

[Eisen 17](#_Toc9779483)

[Scrumboard Eindopdracht 18](#_Toc9779484)

# Functioneel ontwerp

## Opdrachtgever

Drenthe College

Domein D t/m G Level 2:Documentatie Versiebeheer & Scrum

Opdracht: Galgje.

Docenten: Erik Mast, Aminah Balfaqih

## Algemene eisen

Dit zijn de volgende eisen van de opdrachtgever:

*“Om dit level te kunnen halen gaan we een stukje estafette programmeren.*

*De bedoeling is dat jullie na elk stukje programmeren jouw code doorgeeft aan jouw collega student. Deze student gaat dan de code dan verder afmaken. Hierbij is het verboden om de code van de vorige student aan te passen.*

*Natuurlijk moet dit allemaal gepland worden zoals je geleerd hebt bij level 1. Voor eventuele informatie wordt er daarom ook terugverwezen naar de informatie in de DG1 module. Het project dient gemaakt te worden in Java. “*

## Eindgebruiker(s)

## Wat zijn de verwachtingen

*Even de regels van dit project op een rijtje:*

*1. Je programmeert om de beurt in een klasse. Je mag in één beurt maar één methode maken.*

*2. In elke beurt beschrijf je een nieuwe functionaliteit in een taak, deze beschrijving voeg je ook toe aan het FO. Als voor die nieuwe functionaliteit nieuwe klassen noodzakelijk zijn, maak je die aan, maar je vult verder niets in.* ***Let op: geen programmeerwerk in de nieuwe klasse.***

*3. Je zorgt dat je je code werkend aflevert in GitHub, met eventueel een nieuwe taak voor dingen die zijn blijven liggen, of niet goed werken.*

*4. Elke beurt duurt een half uur.*

*5. De volgende beurt pak je een nieuwe taak op.* ***Niet in de klasse waar je teamgenoot in werkt!***

*6. Afronding: Zodra je programma werkt, laat je deze zien aan de docent.*

*a. Bewijs van commits is essentieel voor deze opdracht.* ***Geen Git commits is geen afronding****.*

*b. Het programma moet werken en eventuele resterende bugs, moeten wel in GitHub gedocumenteerd zijn.*

*c. Het FO bevat alle functionaliteit die je bedacht hebt. Het TO bevat alle technische aspecten daarvan dus ook een object model.*

*d. Het object model bevat minstens een klasse voor het scherm, en een voor de logica. “*

# Technisch Ontwerp

## Functionele eisen

Om galgje je te maken in Java hebben we voor de volgende elementen gekozen om te gebruiken:

* Public class
* Public Static void
* Private static
* Array (om alle 30 worden erin te zetten)
* Math random.
* Java.util.Scanner
* While loop
* For loop
* System.out.println();
* Equals ();
* charAt ();
* length ();
* contains ();

## 

## Toelichten

Voor het maken van galgje in het programmeertaal Java zullen we gebruik gaan maken van 4 methodes.

**Public class galgje**

**Public Static void main(String[] args){**

**}**

**Public static void Strop(String raad)**

**Public static void Galg()**

## Eindproduct

## Systeemonderdelen

## Kwaliteitseisen

## Afhankelijkheden

## Structuren

# User-stories

## Eisen

## Scrumboard Eindopdracht